IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant: Sebastien BOUAT,
          et al.   ) Group: Not yet assigned
)
Serial No.: Not yet assigned)
) Examiner: Not yet assigned
Filed: Concurrently herewith)
) Our Ref: B-4343 619198-0
For: "MESSAGE-BASED SOFTWARE)
     SYSTEM"   ) Date: October 29, 2001

CLAIM TO PRIORITY UNDER 35 U.S.C. 119

Commissioner of Patents and Trademarks
Box New Patent Application
Washington, D.C. 20231

Sir:

[X]   Applicants hereby make a right of priority claim under 35

U.S.C. 119 for the benefit of the filing date(s) of the

following corresponding foreign application(s):

COUNTRY	FILING DATE	SERIAL NUMBER
EP	31 October 2000	00410131.7

[ ]   A certified copy of each of the above-noted patent

applications was filed with the Parent Application

No._____.

[X]   To support applicant's claim, a certified copy of the above-

identified foreign patent application is enclosed herewith.

[ ]   The priority document will be forwarded to the Patent Office

when required or prior to issuance.

Respectfully submitted,

Ross A. Schmitt
Attorney for Applicant
Reg. No. 42,529

LADAS & PARRY
5670 Wilshire Boulevard
Suite 2100
Los Angeles, CA 90036
Telephone: (323) 934-2300
Telefax: (323) 934-0202

THIS PAGE BLANK (USPTO)

Eur päisches Patentamt	European Patent Office	Office européen des brevets

Bescheinigung Certificate Attestation

Die angehefteten Unterla-gen stimmen mit der ursprünglich eingereichten Fassung der auf dem näch-sten Blatt bezeichneten europäischen Patentanmel-dung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr.	Patent application No.	Demande de brevet n°
	00410131.7	

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

I.L.C. HATTEN-HECKMAN

DEN HAAG,DEN
THE HAGUE,   13/12/00
LA HAYE,LE

EPA/EPO/OEB Form   1014   - 02.91

Europäisches Patentamt
European Patent Office
Office européen des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.:　　00410131.7
Demande n°:

Anmeldetag:
Date of filing:　31/10/00
Date de dépôt:

Anmelder:
Applicant(s):
Demandeur(s):

HEWLETT-PACKARD COMPANY

Palo Alto, California 94304-1181
UNITED STATES OF AMERICA

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:

Message-based software system

In Anspruch genommene Prioriät(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat:	Tag:	Aktenzeichen:
State:	Date:	File no.
Pays:	Date:	Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:

/

Am Anmeldetag benannte Vertragstaaten:
Contracting states designated at date of filing: AT/BE/CH/CY/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE/TR
Etats contractants désignés lors du depôt:

Bemerkungen:
Remarks:
Remarques:

THIS PAGE BLANK (USPTO)

1

Message-Based Software System

This invention generally relates to logic systems, which use a message-based API, and in particular to telecommunication systems for example such systems used in Internet Protocol networks when transiting voice (voice IP).

As illustrated on figure 1, a typical gatekeeper and service system consists in a gatekeeper unit 100 and a service platform 200 connected together via a suitable interface 300.

The main functions of these two elements are the following ones :

- gatekeeper 100 handles Registration and Admission Service (RAS) messages, which typically consist in requests for obtaining subsequent so-called Q931 intermediate messages tending to establish a voice link between a caller and at least one callee. The RAS messages may be received by gatekeeper unit 100 from terminals of the Internet protocol network and also sent by gatekeeper unit 100 to the terminals. RAS and Q931 messages are defined by a standard called H323 standard.

- the service platform 200 hosts services such as signalling services, billing services, or call diversion services. Each of these services is handled by a specific service unit 210, 220, 230 of the platform 200.

Gatekeeper unit 100 and service platform 200 communicate together by exchanging messages.

These messages are transferred via interface 300, which is an intermediate unit which typically includes an Application Programming Interface (API), i.e. a set of libraries containing specific tools. Such an API is typically a message-based API, which is a message-based set of libraries. A message-based API uses a communication framework and is made of a set of messages conveyed over this framework.

In such known telecommunication systems, gatekeeper unit 100 receives messages from Internet protocol end-points (for example IP phones or Personal computers) through a series of connections 105, which messages each consist in a series of fields arranged together according to a

specific format. This format is typically defined by a known standard called ASN.1.

Such incoming ASN.1 messages generally are in an encoded form under a known encoding standard called PER (Pact encoding Rules), so that they must be decoded by gatekeeper unit, which requires heavy decoding resources in the gatekeeper unit.

For each incoming message, the gatekeeper unit 100 decodes the message and then determines which service unit 210, 220, 230 is the destination unit that requires data contained in this message.

It appears that the data required by service units are different from one service unit to the other. Furthermore, the data are accepted by a given service unit 210, 220, 230, only if those data are under the specific format of the considered unit. For example, unit 210 may require a C data structure and unit 220 may require a XML data structure.

In the known art, the gatekeeper unit 100 sends the whole received message to the service platform 200, which then carries out filtering and formatting of the message upstreams of the service units 210, 220 and 230.

In order to allow those entire ASN.1 messages to be transferred between gatekeeper 100 and service platform 200, it is necessary to encode them back into the PER standard in the gatekeeper unit 100 upstreams of the API 300. Actually, for such complex messages, such encoding back reduces the transfer workload of the API 300. The PER - encoded message is once again decoded in the service platform 200 after the transit through API 300.

The objective problem is to reduce the heavy workload in the known telecommunication system, in particular to reduce the workload due to decoding/encoding.

The invention aims at solving this problem, i.e. at generally proposing a system and a method for a system using a message-based set of libraries which substantially reduces the resources required for conveying the messages so as to improve overall performance and efficiency.

A system according to the invention is a system including a software component comprising input means for receiving messages from other

3

systems, and output means for sending messages to a series of telecommunication service applications, characterised in that the system further includes a message-based set of libraries capable of transmitting messages issued from said output means of the software component to said

5   series of applications, this message-based set of libraries being activated by said software component, in that the software component includes at least two formatter units for formatting messages into at least two different respective message formats for transmission to said applications via said message-based set of libraries.

10   A processing method according to the invention is a method for execution in a gatekeeper and service telecommunication system including a gatekeeper unit which has means for receiving, from an internet protocol network, requests for establishment of communication links, and which gatekeeper unit further has means to send responses to such requests into

15   such a network, the telecommunication system further including a service platform comprising at least two service units, each capable of deriving, from a message received from the gatekeeper unit, service information relating to a communication link to which said message is associated, the service units accepting messages in respective different message formats,

20   and the system further including means for transferring messages from the gatekeeper unit to the service platform and from the service platform to the gatekeeper unit, the method being characterised by the step of formatting messages into said respective message formats of said at least two service units, this formatting step being carried out by at least two formatter units in

25   the gatekeeper unit.

Further features, goals and advantages of the invention will appear to those skilled in the art through the following description, made with reference to the appended figures, in which :

- figure 1 illustrates a prior art telecommunication system ;

30   - figure 2 illustrates a telecommunication system embodying the present invention ;

The telecommunication illustrated on figure 2 (also called telecommunication stack or protocolar stack) consists, similarly to the

4

known stacks, in a gatekeeper unit 100 and a service platform 200 which includes service units 210, 220 and 230. The service units 210, 220 and 230 can be called "users" of the stack.

It also includes an intermediate transfer unit or interface 300 which
5   has the role of transferring messages between the gatekeeper unit 100 and the service platform 200.

This interface here is a message-based Application Programming Interface, which is a message-based set of libraries. Such an API uses a communication framework and is made of a set of messages conveyed over
10   this framework.

As for the known stacks, gatekeeper unit 100 receives messages in an encoded form through connections 105 and decodes those messages into local messages of the gatekeeper unit, in a local language of the gatekeeper unit. The local language is for example language C.

15   The present gatekeeper unit 100 further includes a series of formatter units 110, 120, 130 which are each adapted to produce a preliminary processing of the local messages before sending them through intermediate transfer unit 300. Each unit 110, 120 and 130 transforms local messages into messages of a particular format which complies to the specific
20   requirements of a particular service unit among units 210, 220 and 230.

After decoding an incoming message and determining which service unit 210, 220 or 230 is concerned by the message, gatekeeper 100 transmits the decoded local message to the concerned unit or to the concerned units among units 110, 120 and 130. Units 110, 120 and 130
25   format the local messages into the specific formats of service units 210, 220 and 230 respectively.

When unit 110 receives a local representation of the ASN.1 message, which is constituted of a series of fields, it performs a selection among the fields of this message, and retrieves the only data which are
30   necessary for the corresponding service unit 210 as will be described in greater detail hereinafter. Unit 110 only reads the values of the selected fields of the message and generates a filtered message with said selected values, in the format required by service unit 210.

5

Hence, unit 110 constitutes a filter which produces a simplified or shortened message, transmitted through the interface 300 with lesser band width required and more simple presentation (no heavy PER encoding).

In the present example, interface 300 includes an API. More
5   generally, interface 300 can easily be built, based on a known API model called "Opencall telecommunication stack".

Before transmitting the filtered message through API 300, unit 110 converts the retrieved data into the format of service unit 210, so that service platform 200 receives a message which is ready to be sent to
10   service unit 210.

Service units 210, 220 and 230 may include C++ sets of instructions, Java applets, or programs in languages which are specific to given programming environments such as applications running on top of a platform called Service Execution Platform (SEP) developed by the
15   applicant.

All of these types of applications know different data formats : a C application handles C-structures, a Java applet handles XML structures, SEP platform uses either ASN.1 standard or another known proprietary format called "Data Description Language" (DDL).

20   The gatekeeper unit 100 of the present example can be replaced by any software component exporting a message-based application to some applications, for example any telecommunication software component aiming at authorising or not a communication link to be established.

Actually, the fact that a gatekeeper unit 100 according to the present
25   invention has its own formatter units makes the service platform 200 and the interface 300 independent from the gatekeeper unit.

In other words, service platform 200 and also interface 300 can be adopted with any such software component having the formatting units 110, 120, 130. Actually the platform 200 does not have to perform any message
30   re-formatting work that would be specific to a given software component 100 because message format is the natural format understood by the concerned service unit.

Software component 100 is also independent from the service units 210, 220, 230 and from the message formats that the service units 210, 220, 230 can understand. The gatekeeper unit does not a-priori know about the service units. It is advantageously able to cope with the requirements of any service unit, i.e. to format each message in all the possible formats, each format corresponding to a specific formatter unit. The software component 100 is independent from any limitation of service units regarding data format.

In the present example, it is particularly advantageous that the formatter units 110, 120 and 130 are libraries, the software component 100 links at run-time.

Formatter units 110, 120, 130 will advantageously use an Application and Programming Interface including means for accessing the desired data of the message in the message representation which is local to gatekeeper unit 100.

In the above embodiment, formatter units 110, 120 and 130, perform both a filtering and a conversion of the local messages of gatekeeper unit 100 into languages which are specific to the concerned service units 210, 220, 230, respectively.

It is also possible according to the invention that formatter units realise only translations, or only filtering.

Hereafter are given two examples of formatter units that each perform both filtering and translation. The formatter unit according to Example 1 formats data which are then forwarded to a billing unit. The formatter unit according to Example 2 formats data which are then forwarded to a call diversion unit.

Example 1

A billing service is a service which derives billing information associated with a communication link, on the basis of messages transmitted to this service.

A billing service typically deals with connection establishment and hang-up, in order to compute the duration of a call. In a simple model (fixed pricing) it simply needs the identity of the caller to set up customer's bill.

7

For the illustrative purpose, let us consider that the billing service processes admission (ARQ) and disengage (DRQ) messages complying with the H323 standard.

The ASN.1 structure of an ARQ message is the following:

5   *AdmissionRequest ::=* **SEQUENCE**

{

requestSeqNum	*RequestSeqNum,*
callType	*CallType,*
callModel	*CallModel* **OPTIONAL,**
10 endpointIdentifier	*EndpointIdentifier,* •
destinationInfo	**SEQUENCE OF** *AliasAddress* **OPTIONAL,**
destCallSignalAddress	*TransportAddress* **OPTIONAL,**
destExtraCallInfo	**SEQUENCE OF** *AliasAddress* **OPTIONAL,**
srcInfo	**SEQUENCE OF** *AliasAddress,*
15 srcCallSignalAddress	*TransportAddress* **OPTIONAL,**
bandWidth	*BandWidth,*
callReferenceValue	*CallReferenceValue,*
nonStandardData	*NonStandardParameter* **OPTIONAL,**
callServices	*QseriesOptions* **OPTIONAL,**
20 conferenceID	*ConferenceIdentifier,*
activeMC	*BOOLEAN,*
answerCall	*BOOLEAN,*
...,	
canMapAlias	*BOOLEAN,*
25 callIdentifier	*CallIdentifier,*
srcAlternatives	**SEQUENCE OF** *Endpoint* **OPTIONAL,**
destAlternatives	**SEQUENCE OF** *Endpoint* **OPTIONAL,**
gatekeeperIdentifier	*GatekeeperIdentifier* **OPTIONAL,**
tokens	**SEQUENCE OF** *ClearToken* **OPTIONAL,**
30 cryptoTokens	**SEQUENCE OF** *CryptoH323Token*
OPTIONAL,	
integrityCheckValue	*ICV* **OPTIONAL,**
transportQOS	*TransportQOS* **OPTIONAL,**

8

```
        willSupplyUUIEs          BOOLEAN
}
```

The identity of the caller can be found in the *EndpointIdentifier* field (& 128 character string). Therefore the ARQ message that the H323
5 gatekeeper unit 100 forwards to the billing service unit will hold this single field as a result of a filter function realised by the formatter unit.

A billing service running on an SEP platform of the OpenCall type uses DDL (Data Description Language) for formatting messages. This can be considered as a sub-set of C data structures, so that the representation
10 of the ARQ message will be :

```
struct EndpointIdentifier {
    int size;              // The actual size of the string
    char string[128];      // The content of the string
};
15 struct AdmissionRequest {
    struct EndpointIdentifier endpointIdentifier;
};
```

The formatter unit which produces such an ARQ message can build messages handling directly DDL data as C structures.
20 Another type of billing service may be implemented as an applet running in a Java Virtual Machine and using the XML standard for formatting data. The representation of the ARQ message would be in such case :

```
<xsd:simpleType name="EndpointIdentifier" base="xsd:string">
    <xsd:maxLength value="128"/>
25 </xsd:simpleType>
<xsd:complexType name="AdmissionRequest">
    <xsd:element name="endpointIdentifier" type="EndpointIdentifier"/>
</xsd:complexType>
```

The considered formatter unit can build messages handling XML
30 data through any existing XML engine available as a C library.

As concerns DRQ messages, they are only used as triggers, i.e. the service does not need to know the contents of any field thereof. Therefore,

9

DRQ messages are forwarded by the gatekeeper units 100 as empty messages without representation.

Example 2

Another service such as a call diversion service may be running at
5   the same time as the billing service.

A call diversion is a service which, on the basis of a message transmitted to this service, derives information about an end point with which a communication link should be established, end point which is different from that initially designated in an original link designation. In other
10   words, such a service looks at the called endpoint and forwards the call to another endpoint if the original one is registered for diversion.

Such a service needs the identity of the called endpoint, found from the *destinationInfo* field in an incoming ARQ message. Therefore, the structure of a DDL message to be directed to a call diversion service unit is
15   quite different from the structure of a message sent to a billing service unit since it only contains the set of alias addresses of the called endpoint.

It is thus understood that the billing and the call diversion services use two distinct formatters that construct two different messages, each including a specific part of a same incoming ARQ message.
20   When many service units share the same data format (this applies for instance to OpenCall SEP services that all handle DDL messages), the generation of the formatter units can be automated as will now be described.

A formatter unit generator is provided to the user in the form of a
25   graphical tool, allowing to select the fields and sub-fields each service unit needs in each message. The user also sets constraints on ASN.1 "sequence-of" types (arrays of items).

Sequence-of may be unbound, or upper bound may be high. Custom then need to set a reasonable upper bound.
30   Then the service creation environment has means to generate automatically both the data types required for developing the service, and the formatter that converts messages from the gatekeeper into those data types and vice-versa (typically DDL).

10

Many service units may also share the same formatter unit. For instance a default formatter exporting a reasonable DDL subset of the H323 message-set, could apply to most of the service units running on top of the OpenCall SEP. Other services with specific requirements should use their

5   own formatter.

A formatter unit typically provides an encode/decode interface. It performs the encoding operation before forwarding a message to the service. It also processes messages received from the service unit through the decoding operation.

10   As already described, the formatter unit advantageously makes use of an API to access the local representation (LR) of messages within the gatekeeper unit, i.e. to read some fields of a message which is in a representation used in the gatekeeper. This API, hereafter called local representation API, performs 2 main operations:

15   • Get-field: retrieve the value of a field knowing its logical name (typically its path in the ASN.1 structure).

• Set-field: sets the value of a field knowing its logical name.

Formatter can both format messages directed to the service, and parse messages received from the service. Formatting extracts and presents data

20   from the local representation of a request message so that it can be processed by the service. Parsing extracts data from a reply message and set then in the local representation of this reply.

In addition the Local Representation API should provide operators to know about optional fields, length of arrays, the selected alternative in

25   choices, etc.

It has to be understood that, although the here described embodiement is a gatekeeper system, the invention applies to any software component which exchanges messages with telecommunication service applications and which exports a message based API to some a-priori unknown

30   applications.

It also has to be understood that the different elements described above (service units 210, 220, 230, service platform 200, interface 300, gatekeeper unit 100, formatter units 110, 120, 130) can be physically

11

implemented on one or several hardware equipments being understood that the hardware implementation of the system may be decorrelated from the logic implementation.

THIS PAGE BLANK (USPTO)

12

CLAIMS

1. A system (100, 300) including a software component comprising input means for receiving messages from other systems, and output means

5   for sending messages to a series of telecommunication service applications, characterised in that the system further includes a message-based set of libraries capable of transmitting messages issued from said output means of the software component to said series of applications, this message-based set of libraries being activated by said software component, in that the

10   software component (100) includes at least two formatter units (110, 120, 130) for formatting messages into at least two different respective message formats for transmission to said applications via said message-based set of libraries.

2. A system (100, 300) according to claim 1, characterised in that the

15   message-based set of libraries (300) is a message-based Application Programming Interface.

3. The system (100) of claim 1 or claim 2, characterised in that the software component has means for receiving messages from an internet protocol network and second output means for sending messages into such

20   a network.

4. The system of claim 3, characterised in that said software component forms a gatekeeper component which has means to receive requests for the establishment of communication links from an internet protocol network, and means for sending into the network responses to

25   such requests via said second output means.

5. The system of claim 4, characterised in that it further includes a service platform (200) comprising at least two applications in the form of at least two service units (210, 220, 230), each service unit having means for deriving from a message received from the gatekeeper component (100),

30   service information relating to a communication link to which said message is associated, the service units (210, 220, 230) accepting messages in respective different message formats, and in that said at least two formatter

13

units (110, 120, 130) are capable of formatting messages into said two different respective message formats of said at least two service units.

6. The system of claim 4 or claim 5, characterised in that the gatekeeper component (100) further includes means for decoding messages incoming from an Internet protocol network into a local representation of the gatekeeper component (100).

7. The system of anyone of the previous claims, characterised in that the software component (100) further includes means for dispatching messages onto the formatter units (110, 120, 130), and in that at least one of the formatter units (110, 120, 130) includes filter means for generating a formatted message which includes only part of the data of a message dispatched thereto.

8. The system of claim 7, characterised in that said dispatched messages are in the form of a series of fields and in that said filter means include means to retrieve values of some fields of a dispatched message, and in that the formatter unit which includes the filter means further includes means to produce a message including said retrieved values.

9. The system of any one of the previous claims, characterised in that the formatter units further include means for receiving messages in response to the sent formatted messages, in that the software component (100) includes means for handling messages which consist in a series of fields, and in that at least one of the formatter units (110, 120, 130) includes means for setting a value of a field of a message handled by the software component (100) in accordance with at least one parameter of a received response-message.

10. The system of any one of the previous claims, characterised in that the software component (100) includes means for dispatching messages onto the formatter units (110, 120, 130), and in that at least two formatter units (110, 120, 130) include respective means for converting dispatched messages into two respective different languages.

11. The system of any one of the previous claims, characterised in that said message-based set of libraries is an Application Programming Interface capable of transferring messages which are in different formats.

14

12. The system of anyone of the previous claims, characterised in that said message-based set of libraries is adapted for transmitting differently formatted messages.

13. A method for execution in a gatekeeper and service 5 telecommunication system including a gatekeeper unit (100) which has means for receiving, from an internet protocol network, requests for establishment of communication links, and which gatekeeper unit (100) further has means to send responses to such requests into such a network, the telecommunication system further including a service platform 10 comprising at least two service units (210, 220, 230), each capable of deriving, from a message received from the gatekeeper unit (100), service information relating to a communication link to which said message is associated, the service units (210, 220, 230) accepting messages in respective different message formats, and the system further including 15 means for transferring messages from the gatekeeper unit (100) to the service platform (200) and from the service platform (200) to the gatekeeper unit (100), the method being characterised by the step of formatting messages into said respective message formats of said at least two service units (210, 220, 230), this formatting step being carried out by at least two 20 formatter units (110, 120, 130) in the gatekeeper unit (100).

15

ABSTRACT

Message-Based Software System

5

A system is described including a software component comprising input means for receiving messages from other systems, and output means for sending messages to a series of telecommunication service applications, characterised in that the system further includes a message-based set of

10   libraries capable of transmitting messages issued from said output means of the software component to said series of applications, this message-based set of libraries being activated by said software component. The software component includes at least two formatter units for formatting messages into at least two different respective message formats for transmission to

15   said applications via said message-based set of libraries.
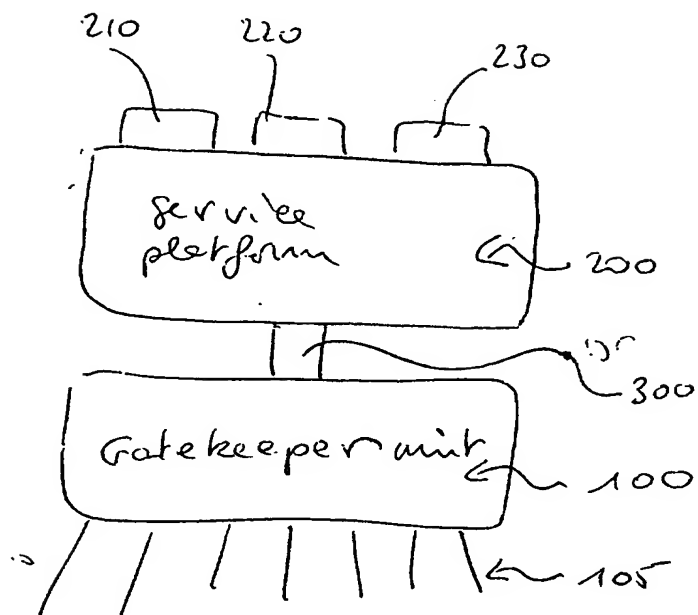
Fig 1

20

THIS PAGE BLANK (USPTO)

1/2

figure 1

2/2

220

230

210

Service Platform

200

300

130

110

Gatekeeper unit

100

120

105